MILATARI
Milwaukee Area Atari Users Group

Vol.5 Nbr.10 | Price $1.50 | September 1986

Calendar of Events

Sept. 16 Beginers SIG at Ron Friedel's
8057 N 45th St. 354-1717
7PM Bring Your Own Questions!

Sept. 20
2:30PM ST SIG
3:30PM Business Meeting
4:15PM Graphic Artist Demo
by Peter Kurth

Sept. 22 Board of Directors Meeting 7:30PM Ground Round Hwy100 & Bluemound

NEW FALL WORKSHOP SCHEDULE ANNOUNCED.....

WHAT: Milatari is please to announce a new schedule of workshops. This new series of workshops will be held once each month. Each workshop session will deal with one topic of interest to club members.

WHEN: Workshops will be held on the first Wednesday of each month. The sessions will begin at 7:30 PM and be over by 9:30 PM.

WHERE: Waukesha State Bank Community Room, 110 Madison Street, Waukesha.

WHY: To provide you with information and support on a timely basis, on informative topics.

HOW: Each month we will bring together the resources from within the Atari community and from the micro-computer industry to make presentations, participate in panel discussions and assist in hands-on workshops.

The schedule for the rest of 1986 is:

October 1st — Telecommunications - A presentation of available telecommunications hardware and software for 8 and 16 bit Ataris. Tutorials on how to access our club's BBS and well as GEnie™ as Compu-Serve™ will be given.

November 5th — Computer languages - A presentation of languages available for the 8 and 16 bit Atari computers. What are their strengths and weeknesses. A panel discussion with the presentors will follow.

December 3rd — Word processors - Hands on training session for the popular word processors on the 8 and 16 bits Atari computers.

Future plans: Spreadsheets, data bases, electronic publishing, etc. If you have any requests, please call Gary or Dave.

FINE TUNING DOS

by Divemaster

[Reprinted from the March edition of the Alamo Area Atari User's Group newsletter with initial attribution to the DIVEMASTER, Sysop of the ATLANTIS BBS - 305/920-6203.]

HOW TO DELETE "TWIN" FILES

Have you ever ended up with two (or more) files on your disk with the same filename? Do you end up cursing and screaming when you try to delete one of them, then find that DOS has deleted BOTH of them? There is a way around that, folks:

1.) Boot up DOS with BASIC in.
2.) In the immediate mode, type POKE 3118,0 [RETURN].
3.) Type DOS, and press [RETURN].

Now you'll be able to delete without losing both files, because by POKEing 3118,0, DOS will erase ONLY the first "twin" file. How about that?

SPEED UP DOS 2.0S

If your DOS 2.0S seems to write very slowly compared to other DOS's, that's 'cause it has a WRITE VERIFY built into it. You can easily create a new version of the DOS without the WRITE VERIFY built in. (Most of the time, you don't need it anyway -- and some other DOS's give you the option of toggling it on and off). Here's what to do:

1.) Boot up DOS with BASIC in.
2.) In the immediate mode, type POKE 1913,80 [RETURN].
3.) Type DOS, and press [RETURN].
4.) Now, re-write DOS to your disk by using the Write DOS and DUP function of your DOS menu.

You have created a new DOS with the WRITE VERIFY turned off. It will write much faster now. COOL?

ARRGH! ERROR 164 AGAIN???!!!!

Awwww. You got an ERROR 164 again?? Well, did you know that you can still force DOS to load in the program anyway? ERROR 164 means that you have screwed-up data on your disk. If you have a LOT of garbage, you'll have a big problem. But what if only a little bit of your file got messed up, and you just KNOW that you could fix it if you could just get the file to load? Well, never fear, do the following:

1.) Boot in DOS with BASIC in.
2.) In the immediate mode, type POKE 4148,234:POKE 4149,234 [RETURN].
3.) Now LOAD your program either from BASIC or from DOS.

You won't get ERROR 164 now. And, you can fix that garbled program .... Ain't that just peachy?

IF YOU HAVE MORE THAN 2 DRIVES

Some DOS's (like SMARTDOS, SpartaDOS, etc.) are built for systems with multiple drives, but other DOS's (Atari 2.0S, M-DOS, etc.) assume that you have only two drives, which is a no-no if you just bought a third drive and your DOS won't access it. Well, all you gotta do is the following:

1.) Boot in DOS with BASIC in.
2.) In the immediate mode, type POKE 1802,15 [RETURN].
3.) Type DOS and press [RETURN].
4.) Re-write DOS to your disk with the DOS menu function that writes DOS and DUP.

Now you'll have written a new DOS that WILL assume that you have as many as four drives. That ought to do ya!!

CHANGE WILDCARD " * " TO ....

If you like to play around with DOS and want to change your wildcard character from " * " to something else (and it's up to you as to just what that something else is), it's really pretty simple. Dig out your ol' manual that shows your ATASCII characters and values. You'll notice that the ATASCII value of the " * " character is 42. Now pick out whatever character you'd like to use instead, and look up its value. For example, suppose you prefer the " + " character. You'll notice that the " + " character has an ATASCII value of 46. To make the change, do the following:

1.) Boot in DOS with BASIC in.
2.) In the immediate mode, type POKE 3783,46 (or the ATASCII value of whatever character you picked) & press [RETURN].
3.) Type DOS and press [RETURN].
4.) Re-write DOS and DUP using the DOS menu function.

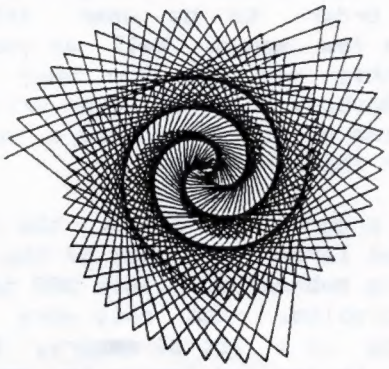
Now your new wild and crazy wildcard character is built right in !!

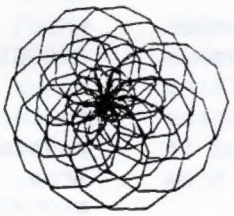
MORE ON PAGE 10

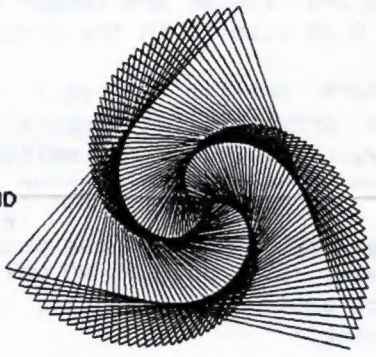
Trigonometric Art With The 1020 Plotter

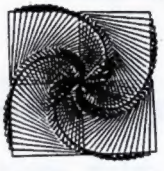
by D. Kramer - BACE

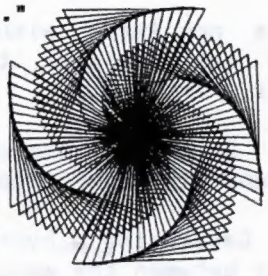
```
100 REM >>                              <<
110 REM >>    BASIC TRIG ART  2.0       <<
120 REM >>                              <<
130 REM >>    by  D. KRAMER  BACE       <<
140 REM >>                              <<
150 REM >>    REQUIRES ATARI 1020       <<
160 REM >>                              <<
170 GOSUB 500
180 REM >>          MAIN LOOP           <<
190 A=R*C
200 XN=X-L*COS(A):YN=Y-L*SIN(A)
210 ? #2;"D";XN;",";YN
220 X=XN:Y=YN
230 FOR Z=1 TO S-1
240 A=A+OA
250 XN=X-L*COS(A):YN=Y-L*SIN(A)
260 ? #2;"D";XN;",";YN
270 X=XN:Y=YN
280 NEXT Z
290 C=C+1:L=L+I
300 IF L>MAXL THEN 330
310 IF PEEK(764)=18 THEN POKE 764,255:GOSUB 420
320 GOTO 190
330 ? "}":? :? :? "PRESS OPTION TO EXIT PROGRAM."
340 ? :? "PRESS START TO RUN PROGRAM AGAIN."
350 IF PEEK(53279)=3 THEN ? #2;"M0,-600*C0":CLOSE #2:END
360 IF PEEK(53279)<>6 THEN 350
370 POKE 53279,7
380 ? #2;"M0,-600"
390 CLOSE #2
400 GOTO 170
410 REM >>          PEN CHANGE          <<
420 ? :? :? "}TO CHANGE COLOR OF PLOT INPUT NEW PEN NUMBER(0-3)."
430 ? :? "PEN COLOR":? "0->BLACK  1->BLUE   2->GREEN  3->RED"
440 ? :? "WHAT IS THE NEW PEN COLOR";:INPUT PC
450 ? #2;"C";PC
460 ? :? :? "CONTINUING THE PLOT..."
470 POKE 764,255
480 RETURN
490 REM >>       INITIALIZATION         <<
500 DEG :TRAP 330
510 ? "}BASIC TRIG ART w/1020 PRINTER-PLOTTER."
520 ? :? "PEN COLOR":? "0->BLACK  1->BLUE   2->GREEN  3->RED"
530 ? :? "TO CHANGE PEN COLOR AFTER START OF    PLOT PRESS 'C' THEN SUPPLY PEN N
UMBER (0-3)."
540 ? :? "WITH WHAT PEN COLOR DO YOU WISH TO    START THE PLOT";:INPUT PC
550 ? "}":? "YOU WILL BE REQUIRED TO INPUT A NUMBEROF SIDES, AN INITIAL SIDE LEN
GTH, A   LENGTH INCREMENT, AND ";
560 ? " A ROTATIONAL   ANGLE."
570 ? :? "I RECOMMEND:":? " SIDES  1-10":? " LENGTH  -2 TO 2":? " INCREMENT -2 T
O 2"
580 ? " ROTATIONAL ANGLE  20,30,45,60,72,90,   120,144,180 PLUS OR MINUS 0 TO 3
    DEGREES"
590 ? :? "WHAT ARE YOUR INPUT VALUES(S,L,I,R)";:INPUT S,L,I,R
600 LPRINT "S","L","I","R"
610 LPRINT S,L,I,R
620 OA=360/S:MAXL=210-S*17
630 X=240:Y=-240:A=0:C=1
640 OPEN #2,8,0,"P:":? #2;""
650 ? #2;"C";PC
660 ? #2;"M";X;",";Y
670 ? :? "PLOTTING..."
680 RETURN
```

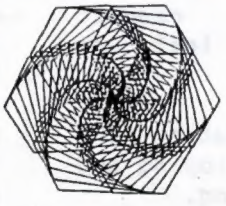


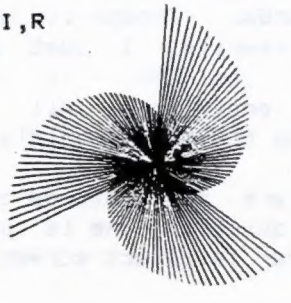












Direct Screen Writing

By Frank Daniel
Reprinted from SLCC Journal

Some years ago I was writing a program that in order to be user friendly required a few menus. Well as you can imagine, this was not too much of a problem. Anybody that has ever written a multi-tasking utility has used a menu at one time or another.

But as the program got larger, the number of menus got larger and most of the menus were getting sub-menus. This WAS getting to be a problem. Not only were these menus taking up a lot of memory, but it was taking longer and longer to get from point A to point B in the program.

Now there are two things I really hate. One is programs that gobble up too much memory. The other is waiting for the program to finish printing a menu. I faced a real dilemma. It is bad enough having just one of these problems in my programs. But both?? NO WAY!! My self respect could not take it. I had to do something!!

I was now faced with three options if I were to continue the project. These were:

1. Doing a complete rewrite of the command processor into a CPM type system.

2. Developing a hybrid which would be a cross between the menu system and the CPM system.

3. Find a way to change the menus fast.

I had to rule out the first option right off the top. A major rewrite just could not be done in the time available. The second option went very quickly afterwards. Though it would not mean a major rewrite, I just could not bring myself to do it (self respect again...drat it). All that left was changing the menus rapidly.

There are two methods of updating the screen quickly. One is page flipping and the other is direct screen writing.

Page flipping is the fastest method of changing the display known to an ATARI programmer. Just change two bytes in the display list and the whole screen changes. But with all this speed comes a few problems.

One, you have to preset all of your menus. By that I mean you have to make sure that all of the characters in the menu have been offset correctly. This is because ANTIC has its own set of character values which are very different from ASCII or ATASCII. Another problem is arranging the menus in memory. The ANTIC is a bit touchy about which page boundaries get passed when describing the screen's data area. (Hint: NEVER-EVER try to pass a 4K boundary!!)

The worst problem with page flipping though was the amount of memory it would use. The "GRAPHICS 0" screen mode uses 960 bytes for it Load Memory Scan (LMS) or data area. Page flipping would require that a number of blocks this size be set aside for the exclusive use of the menu driver. Add to this the dead areas between the menus needed to prevent page boundary problems and you can easily see that the amount of usable program memory is quickly diminished.

That leaves direct screen writes. Doing direct writes also has its problems. The first, like page-flipping, is that most of the text requires offsetting. This is not a big problem. There are many ways to rectify this. You can precalculate the offsets or write a short program that does it for you. Another method is to include an offset routine in the program. This is not very efficient for a menu driver but is very necessary when the text is varied or unknown. I actually use this method in the preceding demo.

Another problem with direct writes is parameter passing. How do you tell the routine where the text is, how long it is and where on the screen to put it. The solution to this is also varied. The first that comes to mind is reserving a place in memory for the parameters. When dealing with BASIC though, it is easier to use the stack.

```
A=USR(CDE,X,Y,ADR(A$),LEN(A$))
```

or the alternate

```
A=(CDE,ADR(A$),LEN(A$))
```

CDE is the address of the machine code string. X is the screen column and Y is the row position where you want to start the display. A$ is the string to be displayed. If you do not pass an X and Y parameter, the routine assumes that you want to use the present cursor position as default.

The routine will display all characters with one exception. The EOL character (155) is used as a line delimeter. This for multi-line displays without the need of counting the characters. To go to the next line, simply insert an EOL character at the appropriate location in the string. One warning, the routine assumes that your starting column position is the left margin and will start from there.

While we are on the subject of warnings, let me caution you about a few items.

First, with the exception of screen position, the routine DOES NOT do any error checking. You can pass strings larger that the screen. This normally is not a problem, but forlorn is the person who does this with a relocated display list and no backup!

If you do make an error in the screen position, the value 141 (cursor out of range) will be passed back.

Do not let the machine code call a subroutine. BASIC gets a little confused. The program does not crash, but does not work right.

Tell us what you do with your RAM. Write an article today!

```
1 REM ** SAN LEANDRO COMPUTER CLUB **
10 CLR
20 POKE 82,0:POKE 83,40:POKE 752,1:? "
ß"
30 DIM A$(960),CODE$(196)
40 REM ** CODE LODING ROUTING  **
70 POSITION 13,2:? "LOADING CODE":FOR
I=1 TO 195:READ A:CODE$(I,I)=CHR$(A):N
EXT I
80 CDE=ADR(CODE$)
90 REM ** INITIALIZE TEST STRING **
110 POSITION 10,4:? "INITIALIZING STRI
NG":FOR J=1 TO 40:FOR I=0 TO 23:K=I*40
+J:A$(K,K)=CHR$(64+J)
120 NEXT I:NEXT J
130 REM
140 REM ** SHOW SPEED DIFFERENCE BETWE
EN A PRINT AND A DIRECT WRITE **
150 REM
160 ? "ß":POSITION 12,2:? "FIRST USE A
 PRINT":POKE 20,0
170 IF PEEK(20)<100 THEN 170
180 ? A$;:POKE 20,0
190 IF PEEK(20)<50 THEN 190
200 ? :? "                NOW THE FAST WA
Y"
210 FOR I=0 TO 1000:NEXT I
220 FOR I=0 TO 24:? :NEXT I:POSITION 0
,20
230 A=USR(CDE,0,0,ADR(A$),LEN(A$))
240 FOR I=0 TO 500:NEXT I
250 ? "ß":? "      NOW FOR A LITTLE FUN
 WITH";:POKE 20,0
260 IF PEEK(20)<100 THEN 260
280 REM ** SHOW POSITIONAL CAPABILITIE
S **
300 A$="":A$=" FAST RANDOM screen acce
ss ":A$(27,27)=CHR$(30):A$(1,1)=CHR$(3
1)
310 A=USR(CDE,5,2,ADR(A$),LEN(A$)):POK
E 20,0
320 IF PEEK(20)<150 THEN 320
330 FOR I=10 TO 100:X=RND(0)*40:Y=RND(
0)*24:A=USR(CDE,X,Y,ADR(A$),LEN(A$))
340 NEXT I
350 FOR I=0 TO 150:NEXT I:? "ß":POSITI
ON 14,2:? "END OF DEMO":END
370 REM ** DATA FOR MACHINE CODE **
390 DATA 169,0,162,5,149,203,202,16,25
1,104,201,2,208,6,164,84,166,85,208,27
,104,104,170,104
400 DATA 104,168,224,41,16,4,192,24,14
4,13,104,104,104,104,169,141,133,212,1
69,0,133,213,96,169
410 DATA 0,136,48,11,24,105,40,133,207
,144,246,230,208,208,242,165,207,24,10
1,88,133,207,165,208
420 DATA 101,89,133,208,138,24,101,207
,133,207,144,2,230,208,104,133,204,104
,133,203,104,133,209,104
430 DATA 170,208,2,198,209,165,207,133
,205,165,208,133,206,160,0,177,203,201
,155,240,49,72,41,127
440 DATA 201,32,16,6,104,24,105,64,144
,11,201,96,16,6,104,56,233,32,176,1,10
4,145,205,200
450 DATA 208,4,230,204,230,206,202,208
,214,198,209,16,210,169,1,133,212,169,
0,133,213,96,202,208,4,198,209
460 DATA 48,240,165,207,24,105,40,133,
207,144,2,230,208,152,56,101,203,133,2
03,144,165,204,208,161
```

Worldwide Users Network

Permission to reprint or excerpt is granted only if the following line appears at the top of the article:

ANTIC PUBLISHING INC.,COPYRIGHT 1986.
REPRINTED BY PERMISSION.

PROFESSIONAL GEM By Tim Oren
Column #9 - VDI Graphics: Lines and Solids

This issue of ST PRO GEM is the first in a series of two which will explore the fundamentals of VDI graphics output. In this installment, we will take a look at the commands necessary to output simple graphics such as lines, squares and circles as well as more complex figures such as polygons. The following episode will take a first look at graphics text output, with an emphasis on ways to optimize its drawing speed.

A BIT OF HISTORY

One of the reasons that the VDI can be confusing is that drawing anything at all, even a simple line, can involve setting around four different VDI parameters before making the draw call! (Given the state of the GEM documents, just FINDING them can be fun!) Looking backwards a bit sheds some light on why the VDI is structured this way, and also gives us a framework for organizing a discussion of graphics output.

The GEM VDI closely follows the so-called GKS standard, which defines capabilities and calling sequences for a standardized graphic input/output system. GKS is itself an evolution from an early system called "Core". Both of these standards were born in the days when pen plotters, vectored graphics displays, and minicomputers were the latest items. So, if you wonder why setting the drawing pen color is a separate command, just think back a few years when it actually meant what it says! (The cynical may choose instead to ponder the benefits of standardization.)

When doing VDI output, it helps if you pretend that the display screen really is a plotter or some other separate device, which has its own internal parameters which you can set up and read back. The class of VDI commands called Attribute Functions let you set the parameters. Output Functions cause the "device" to actually draw someone once it is configured. The Inquire Functions let you read back the parameters if necessary.

There are two parameters which are relevant no matter what type of object you are trying to draw. They are the writing mode and the clipping rectangle. The writing mode is similar to that discussed in the column on raster operations. It determines what effect the figure you are drawing will have on data already on the screen. The writing mode is set with the call:

```
vswr_mode(vdi_handle, mode);
```

(Vdi_handle, here and below, is the handle obtained from graf_handle at the beginning of the program. Mode is a word which may be one of:

1 - Replace Mode
2 - Transparent Mode
3 - XOR mode
4 - Reverse Transparent Mode

In replace mode, whatever is on the screen is overwritten. If you are writing characters, this means the background of each character cell will be erased.

In transparent mode, only the pixels directly under the "positive" part of the image, that is, where 1-bits are being written, will be changed. When writing characters, the background of the cell will be left intact.

In XOR mode, an exclusive or is performed between the screen contents and what is being written. The effect is to reverse the image under areas where a 1-bit occurs.

Reverse transparent is like transparent, but with a "reverse color scheme". That is, only places where a 0-bit is to be put are changed to the current writing color. When you write characters in reverse transparent (over white), the effect is reverse video.

The other common parameter is the clipping rectangle. It defines the area on the screen where the VDI is permitted to draw. Any output which would fall outside of this area is ignored; it is effectively a null operation. The clip rectangle is set with the call:

```
vs_clip(vdi_handle, flag, pxy);
```

Pxy is a four-word array. Pxy[0] and pxy[1] are the X and Y screen coordinates, respectively, of one corner of your clipping rectangle. Pxy[2] and pxy[3] are the coordinates of the diagonally opposite corner of the rectangle. (When working with the AES, use of a GRECT to define the clip is often more convenient. The routine set_clip() in the download does this.)

Flag is set to TRUE if clipping is to be used. If you set it to FALSE, the entire screen is assumed to be fair game.

Normally, you should walk the rectangle list for the current window to obtain your clipping rectangles. (See PRO GEM #2 for more details.) However, turning off the clip speeds up all output operations, particularly text. You may do this ONLY when you are absolutely certain that the figure you are drawing will not extend out of the top-most window, or out of a dialog.

Extravaganza

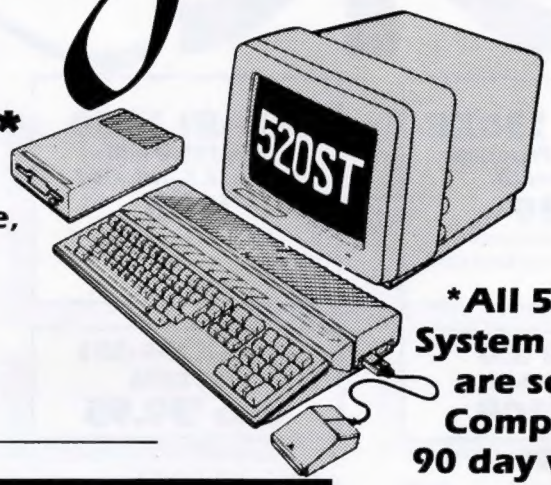
Ad Reference Number... 4U...

ATARI 520 ST SYSTEM PACKAGE*

*Including RGB or Monochrome Monitor, Mouse, Disk Drive, Basic, Logo, Neochrome, 1st Word, TOS on ROM, and RF Modulator

$649 MONOCHROME SYSTEM

$769 RGB/COLOR SYSTEM



***All 520ST System Packages are sold with a ComputAbility 90 day warranty.**

520ST ONE MEGABYTE UPGRADE

Computer must be sent in to service center **$175**

ATARI SF 314 DISK DRIVE

Double sided/ 1 Megabyte storage **$209**

MichTron 520 ST SUPER SPECIALS*

Major Motion	24.95	Cornerman	31.95
Time Bandit	24.95	D.F.T.	31.95
Goldrunner	24.95	The Animator	24.95
Solitaire	24.95	Mictron Util.	37.95
Kissed	31.95	Mi-Term	31.95
DOS Shell	24.95	Softspool	24.95

*Prices effective now through August 31, 1986

SUPRA HARD DRIVES

10 Megabyte	649
20 Megabyte	849
30 Megabyte	Call
60 Megabyte	Call

QMI ST 1200 BAUD MODEM

Direct connect with ST talk **$159**

ATARI 520 ST • ATARI 1040 ST SOFTWARE

INFOCOM ST

Forever Voyaging	29.95
Ballyhoo	27.95
Cutthroats	27.95
Deadline	34.95
Enchanter	27.95
Hitchiker	27.95
Infidel	29.95
Planetfall	27.95
Sea Stalker	27.95
Sorcerer	29.95
Spellbreaker	34.95
Starcross	34.95
Suspect	29.95
Suspended	34.95
Wishbringer	27.95
Witness	27.95
Zork I	27.95
Zork II or III	29.95

OSS ST

Personal Pascal	49.95
Disk Kit	27.95
Personal Prolog	79.95

ST BUSINESS

VIP Professional	119.95
Synsoft Gen. Ledger	84.95
SBM (Point of Sales)	84.95
Cash Disbursements	69.95
Sierra Accts. Rec.	69.95
Sierra Gen. Ledger	69.95
Sierra Payroll	69.95
VIP Lite	67.95
Financial Cookbook	34.95

MICHTRON

Alt	34.95
BBS	34.95
Business Tools	34.95
Calender	20.95
Cornerman	34.95
D.F.T.	34.95
DOS Shell	34.95
Echo	27.95
Flipside	27.95
Goldrunner	27.95
Kissed	34.95
Lands of Havoc	16.95
M-Copy	54.95
M-Disk	27.95
Major Motion	27.95
Mi-Term	34.95
Michtron Utilities	39.95
Mudpies	27.95
Softspool	27.95
Solitaire	27.95
The Animator	27.95
Time Bandits	27.95

HIPPOPOTAMUS

Almanac	23.95
Disk Utilities	34.95
Eprom Burner	99.95
Jokes & Quotes	23.95
Ramdisk	23.95
Backgammon	27.95
Hippoconcept	62.95
Hippopixel	27.95
Hipposimple	34.95
Hippospell	27.95
Hippoword	62.95
X-10 Powerhouse	Call

ST WORD PROCESSORS

Final Word	94.95
Express	31.95
Paperclip Elite	Call
Regent Word	31.95
Regent Spell	31.95
Habawriter	54.95

ST DATABASES

DB Man	69.95
H&D Base	64.95
Zoomracks	59.95

ST GRAPHICS

Degas	27.95
N-Vision	27.95
Easy Draw	99.95
PC Board Designer	249

ST UTILITIES

ST Toolbox	24.95
Macrodesk	20.95
Deskmaster	27.95
ST Forth	31.95
Music Studio	39.95
Sound Wave	33.95
Abacus Books	Call

EST. 1982

ComputAbility™

P.O. Box 17882, Milwaukee, WI 53217

ORDER LINES OPEN

Mon-Fri. 11 a.m. - 7 p.m. CST • Sat. 12 p.m. - 5 p.m. CST

To Order Call Toll Free

800-558-0003

For Technical Info, Order Inquiries, or for Wisc. Orders

414-351-2007

DISKETTES

Call for our current low disk prices on Sony & Maxell 3.5 Diskettes

ST ACCESSORIES

Flip n' File II-Micro	19.95
Dustcovers	Call
3.5 Disk Drive Clean Kit	Call
Mouse Pad	8.95
Mouse House	6.95

ST ADVENTURES

Ultima II	39.95
Crimson Crown	27.95
Farenheit 451	33.95
Transylvania	27.95
Treasure Island	27.95
King's Quest II	33.95
Perry Mason	33.95
Word Invaders	22.95
Borrowed Time	33.95
Hacker	29.95
9 Princes/Amber	33.95
Sundog	24.95
Mindshadow	33.95
Winnie The Pooh	19.95
Black Cauldron	27.95
Amazon	33.95
Spiderman	16.95
Fantastic Four	16.95
Sword of Kadash	27.95
Apshai Trilogy	27.95
Universe II	47.95
The Pawn	29.95
Ultima III	39.95

ST TELE-COMMUNICATION

PC Intercom	84.95
ST Talk	17.95

ST PRINT UTILITIES

Typesetter	24.95
Rubber Stamp	24.95
Printmaster	27.95
Art Gallery I	20.95
Fontwriter	27.95

ST ARCADE GAMES

Hex	27.95
Monkey Business	16.95
Delta Patrol	16.95
Compubridge	19.95
Bridge 4.0	20.95
Winter Games	27.95
Rogue	27.95
Diablo	20.95
Super Huey	27.95
Phantasie	27.95
Mean 18	34.95
Leader Board	27.95
Brattacus	33.95
Donald Duck	20.95
Silent Service	Call
Flight Simulator II	Call

ATARI is a trademark of ATARI, INC.

No surcharge for MasterCard or Visa

ORDERING INFORMATION: Please **specify system.** For fast delivery send cashier's check or money order. Personal and company checks allow 14 business days to clear. School P.O.'s welcome. **C.O.D. charges are $3.00.** In Continental U.S.A. include $3.00 for software orders. 4% shipping for hardware, minimum $4.00. Master Card and Visa orders please include card #, expiration date and signature. WI residents please include 5% sales tax. HI, AK, FPO, APO, Puerto Rico and Canadian orders please add 5% shipping, **minimum $5.00.** All other **foreign orders add 15% shipping, minimum $10.00.** All orders shipped outside the Continental U.S.A. are shipped first class insured U.S. mail. If foreign shipping charges exceed the minimum amount, you will be charged the additional amount to get your package to you quickly and safely. All goods are new and include factory warranty. Due to our low prices all sales are final. **All defective returns must have a return authorization number.** Please call (414) 351-2007 to obtain an R.A. # or your return will not be accepted. Prices and availability subject to change without notice.

Atari

ATARI 130XE
& 2 FREE Programs our choice
$139
This is shipped price anywhere in Continental USA

ATARI 1050
Disk Drive with D.O.S. 2.5 included
$135
This is shipped price anywhere in Continental USA

star micronics inc
NX-10 PRINTER & U-Print A Interface
$299
This is shipped price anywhere in Continental USA

130 XE COMPUTER PKG.
- 130XE Computer
- 1050 Disk Drive
- 1027 Printer
- Atariwriter Plus
- Star Raiders
- Paint
- Music Painter
- E.T.
- Timewise

$395
This is shipped price anywhere in Continental USA

ATARI 850 INTERFACE ONLY $109

ATARI XM-301 MODEM ONLY $39.95

ATARI 1027 Printer & ATARIWRITER PLUS $129

PANASONIC 1091 & U-PRINT A INTERFACE
$299
This is shipped price anywhere in Continental USA

MONITORS

Gold Star Amber	79.95
Commodore 1802	179
NEC 1225 Color	139
Amdek	Call
NAP Amber	89.95

Monitors Shipping $10

PRINTER BUFFERS

U-Buff 16K	79.95
U-Buff 64K	99.95

MODEMS

Prometheus 1200	299
Maxwell 1200	229
Supra 1200 AT	169
Atari XM-301	39.95
Supra 300 AT	44.95

SUPER SPECIALS*

Buy any two software programs for the Atari 130XE or Atari 800/800XL at our everyday low price and choose one of these programs ABSOLUTELY FREE
(while supplies last)

TIMEWISE
E.T. PHONE HOME
STAR RAIDERS
MUSIC PAINTER
OLLIES FOLLIES
ATARI PAINT
KRAZY SHOOTOUT

*Prices effective now through August 31, 1986

star micronics inc

NX-10	Call	SD-15	449
NL-10	Call	SR-10	Call
SG-15	369	SR-15	Call
SD-10	339		

PRINTER INTERFACES

U-Print A	49.95
MPP 1150	54.95
P.R. Connection	59.95

ATARI 800 • 800 XL • 1200 XL • 130 XE SOFTWARE

ACTIVISION

Designer Pencil	17.95
Cross Country Race	17.95
Hacker	17.95
Mindshadow	17.95
Music Studio	19.95
Space Shuttle	17.95

BRODERBUND

Karateka	20.95
Champ Loderunner	23.95
Print Shop	28.95
Bank Street Writer	34.95
Print Shop Graphics I, II or III	19.95
Print Shop Paper	16.95
Prt. Shop Companion	27.95

INFOCOM

Deadline	29.95
Enchanter	24.95
Infidel	24.95
Planetfall	24.95
Sorcerer	24.95
Starcross	29.95
Suspended	27.95
Witness	29.95
Sea Stalker	24.95
Cutthroats	24.95
Suspect	27.95
Hitchhiker	24.95
Zork I	24.95
Zork II or III	27.95
Wishbringer	27.95
Spellbreaker	29.95
Ballyhoo	27.95
Fooblitsky	27.95

MICROPROSE

Silent Service	23.95
Gunship	23.95
Accrojet	23.95
F-15 Strike Eagle	23.95
Decision in Desert	27.95
Kennedy Approach	23.95
Crusade in Europe	27.95
Conflict/Vietnam	27.95

OPTIMIZED SYSTEMS

Basic XE	49.95
MAC 65XL	47.95
Action	47.95
Basic XL	39.95
All Tool Kits	19.95

ELECTRONIC ARTS

Archon	19.95
Archon II	24.95
Mule	19.95
Realm/Impossibility	19.95
Murder/Zinderneuf	19.95
Music Construction	19.95
Pinball Construction	19.95
One on One	24.95
Seven Cities of Gold	24.95
Financial Cookbook	29.95
Racing Destruction	24.95
Super Boulderdash	19.95
Chessmaster 2000	27.95

GAMESTAR

Star League Baseball	17.95
Starbowl Football	17.95
On Track Racing	17.95

EPYX

Rescue on Fractalus	24.95
The Eidolon	24.95
Koronis Rift	24.95
Ballblazer	24.95
Summer Games	24.95
World Karate	20.95

XLENT SOFTWARE

Megafont II	17.95
Page Designer	21.95
Typesetter	24.95
Megafiler	21.95
Rubber Stamp	21.95

BATTERIES INCLUDED

Home Pak	34.95
Paper Clip	39.95
B-Graph	34.95

DATASOFT

Alternate Reality	24.95
Zorro	18.95
Goonies	18.95
Neverending Story	18.95
Conan The Barbarian	18.95
Bruce Lee	18.95
Mind Pursuit	18.95
Nibbler	18.95

EST. 1982
ComputAbility™

P.O. Box 17882, Milwaukee, WI 53217
ORDER LINES OPEN
Mon-Fri. 11 a.m. - 7 p.m. CST • Sat. 12 p.m. - 5 p.m. CST

To Order Call Toll Free
800-558-0003

For Technical Info, Order Inquiries, or for Wisc. Orders
414-351-2007

MISCELLANEOUS

TAC III Joystick	14.95
TAC II Joystick	12.95
Starfighter Joystick	9.95
Silk Stik Joystick	7.95
Wico 3-Way Joystick	21.95
Mastertype	27.95
Flight Simulator	34.95
Home Accountant	49.95
Monkey Wrench-Cart	23.95
Sargon III	34.95
Spy vs. Spy	23.95
Odesta Chess	49.95
Ramrod XL	69.95
Universe	69.95
Beachead	21.95
Letter Perfect	39.95
Data Perfect	39.95
Data Perfect	39.95
Fleet System II	49.95
Strip Poker	23.95
Halley Project	23.95
Micro League Base	29.95
Harcourt/Brace S.A.T.	49.95
Ultima II	37.95
Ultima III	34.95
Ultima IV	41.95
Spy Hunter	29.95
Omnimon	69.95
Island Caper	23.95
General Magr./MLB	29.95
Fight Night	19.95
Hardball	19.95
Raid Over Moscow	23.95
Beachead II	23.95
Star Fleet I	34.95

SSI

Carrier Force	37.95
Combat Leader	24.95
Cosmic Balance	24.95
Cosmic Balance II	24.95
Broadsides	24.95
War in Russia	49.95
50 Mission Crush	24.95
Questron	32.95
Rails West	24.95
Computer Ambush	37.95
Computer Baseball	24.95
Reforger 88	37.95
Breakthru/Ardennes	37.95
Field of Fire	24.95
Imperium Galatium	24.95
Oper. Market Garden	32.95
Kampfgruppe	37.95
Comp. Quarterback	24.95
Colonial Conquest	24.95
Gemstone Warrior	21.95
Six Gun Shootout	24.95
Battle of Antietam	32.95
USAAF	37.95
Nam	27.95
Panzer Grenider	24.95
Mech Brigade	39.95
Wizard's Crown	27.95

SYNAPSE

Syncalc	32.95
Synfile	32.95
Loderunner's Rescue	20.95
Syncalc Templates	16.95
Essex	27.95
Mindwheel	27.95
Brimstone	27.95

No surcharge for MasterCard or Visa

ORDERING INFORMATION: Please **specify system.** For fast delivery send cashier's check or money order. Personal and company checks allow 14 business days to clear. School P.O.'s welcome. **C.O.D. charges are $3.00.** In Continental U.S.A. include $3.00 for software orders. 4% shipping for hardware, minimum $4.00. Master Card and Visa orders please include card #, expiration date and signature. WI residents please include 5% sales tax. HI, AK, FPO, APO, Puerto Rico and Canadian orders, please add 5% shipping, **minimum $5.00.** All other **foreign orders add 15% shipping, minimum $10.00.** All orders shipped outside the Continental U.S.A. are shipped first class insured U.S. mail. If foreign shipping charges exceed the minimum amount, you will be charged the additional amount to get your package to you quickly and safely. All goods are new and include factory warranty. Due to our low prices all sales are final. **All defective returns must have a return authorization number.** Please call (414) 351-2007 to obtain an R.A. # or your return will not be accepted. Prices and availability subject to change without notice.

THE LINE FORMS ON THE LEFT

The VDI line drawing operations include polyline, arc, elliptical arc, and rounded rectangle. I'll first look at the Attribute Functions for line drawing, then go through the drawing primitives themselves.

The most common used line attributes are color and width. The color is set with:

vsl_color(vdi_handle, color);

where color is one of the standard VDI color indices, ranging from zero to 15. (As discussed in column #6, the color which actually appears will depend on the pallette setting of your ST.)

The line width may only be set to ODD positive values, for reasons of symmetry. If you try to set an even value, the VDI will take the next lower odd value. The call is:

vsl_width(vdi_handle, width);

The two less used line parameters are the end style and pattern. With the end style you can cause the output line to have rounded ends or arrowhead ends. The call is:

vsl_ends(vdi_handle, begin_style, end_style);

Begin_style and end_style are each words which may have the values zero for square ends (the default), one for arrowed ends, or two for rounded ends. They determine the styles for the starting and finishing ends of the line, respectively.

The line pattern attribute can select dotted or dashed lines as well as more complicated patterns. Before continuing, you should note one warning: VDI line output DOES NOT compensate for pixel aspect ratio. That is, the dashes on a line will look twice as long drawn vertically on a medium-res ST screen as they do when drawn horizontally. The command for setting the pattern is:

vsl_type(vdi_handle, style);

Style is a word with a value between 1 and 7. The styles selected are:

1 - Solid (the default)
2 - Long Dash
3 - Dot
4 - Dash, Dot
5 - Dash
6 - Dash, Dot, Dot
7 - (User defined style)

The user defined style is determined by a 16-bit pattern supplied by the application. A one bit in the pattern turns a pixel on, a zero bit leaves it off. The pattern is cycled through repeatedly, using the high bit first. To use a custom style, you must make the call:

vsl_udsty(vdi_handle, pattern);

before doing vsl_type().

As I mentioned above, the line type Output Functions available are polyline, circular and elliptical arc, and rounded rectangle. Each has its own calling sequence. The call for a polyline is:

v_pline(vdi_handle, points, pxy);

Points tells how many vertices will appear on the polyline. For instance, a straight line has two vertices: the end and the beginning. A closed square would have five, with the first and last identical. (There is no requirement that the figure described be closed.)

The pxy array contains the X and Y raster coordinates for the vertices, with a total of 2 * points entries. Pxy[0] and pxy[1] are the first X-Y pair, and so on.

If you happen to be using the XOR drawing mode, remember that drawing twice at a point is equivalent to no drawing at all. Therefore, for a figure to appear closed in XOR mode, the final stroke should actually stop one pixel short of the origin of the figure.

You may notice that in the GEM VDI manual the rounded rectangle and arc commands are referred to as GDPs (Generalized Drawing Primitives). This denotation is historical in nature, and has no effect unless you are writing your own VDI bindings.

The rounded rectangle is nice to use for customized buttons in windows and dialogs. It gives a "softer" look to the screen than the standard square objects. The drawing command is:

v_rbox(vdi_handle, pxy);

Pxy is a four word array giving opposite corners of the rectangle, just as for the vs_clip() call. The corner rounding occurs within the confines of this rectangle. Nothing will protrude unless you specify a line thickness greater than one. The corner rounding is approximately circular; there is no user control over the degree or shape of rounding.

Both the arc and elliptical arc commands use a curious method of specifying angles. The units are tenths of degrees, so an entire circle is 3600 units. The count starts at ninety degrees right of vertical, and proceeds counterclockwise. This means that "3 o'clock" is 0 units, "noon" is 900 units, "9 o'clock" is 1800 units, and 2700 units is at "half-past". 3600 units take you back to "3 o'clock".

The command for drawing a circular arc is:

v_arc[vdi_handle, x, y, radius, begin, end];

X and y specify the raster coordinates of the center of the circle. Radius specifies the distance from center to all points on the arc. Begin and end are angles given in units as described above, both with values between 0 and 3600. The drawing of the arc ALWAYS proceeds counterclockwise, in the direction of increasing arc number. So values of 0 and 900 for begin and end would draw a quarter circle from "three o'clock" to "noon". Reversing the values would draw the other three quarters of the circle.

A v_arc[] command which specifies a "full turn" is the fastest way to draw a complete circle on the screen. Be warned, however, that the circle drawing algorithm used in the VDI seems to have some serious shortcomings at small radii! You can experiment with the CIRCLE primitive in ST Logo, which uses v_arc[], to see what I mean.

Notice that if you want an arc to strike one or more given points on the screen, then you are in for some trigonometry. If your math is a bit rusty, I highly recommend the book "A Programmer's Geometry", by Bowyer and Woodwark, published by Butterworths [London, Boston, Toronto].

Finally, the elliptical arc is generated with:

v_ellarc[vdi_handle, x, y, xrad, yrad, begin, end];

X, y, begin, and end are just as before. Xrad and yrad give the horizontal and vertical radii of the defining ellipse. This means that the distance of the arc from center will be yrad pixels at "noon" and "half-past", and it will be xrad pixels at "3 and 9 o'clock". Again, the arc is always drawn counterclockwise.

There are a number of approaches to keeping the VDI's attributes "in sync" with the actual output operations. Probably the LEAST efficient is to use the Inquire Functions to determine the current attributes. For this reason, I have omitted a discussion of these calls from this column.

Another idea is to keep a local copy of all significant attributes, use a test-before-set method to minimize the number of Attribute Functions which need to be called. This puts a burden on the programmer to be sure that the local attribute variables are correctly maintained. Failure to do so may result in obscure drawing bugs. If your application employs user defined AES objects, you must be very careful because GEM might call your draw code in the middle of a VDI operation [particularly if the user defined objects are in the menu].

Always setting the attributes is a simplistic method, but often proves most effective. The routines pl_perim[] and rr_perim[] in the download exhibit this approach. Modification for other primitives is straightforward. This style is most useful when drawing operations are scattered throughout the program, so that keeping track of the current attribute status is difficult. Although inherently inefficient, the difference is not very noticable if the drawing operation requested is itself time consuming.

In many applications, such as data graphing programs or "Draw" packages, the output operations are centralized, forming the primary functionality of the code. In this case, it is both easy and efficient to keep track of attribute status between successive drawing operations.

SOLIDS

There are a wider variety of VDI calls for drawing solid figures. They include rectangle or bar, disk, pie, ellipse, elliptical pie, filled rounded rectangle, and filled polygonal area. Of course, filled figure calls also have their own set of attributes which you will need to set.

The fill color index determines what pen color will be used to draw the solid. It is set with:

vsf_color[vdi_handle, color];

Color is just the same as for line drawing. A solid may or may not have a visible border. This is determined with the call:

vsf_perimeter[vdi_handle, vis];

Vis is a Boolean. If it is true, the figure will be given a solid one pixel outline in the current fill color index. This is often useful to improve the appearance of solids drawn with a dithered fill pattern. If vis is false, then no outline is drawn.

There are two parameters which together determine the pattern used to fill your figure. They are called interior style and interior index. The style determines the general type of fill, and the index is used to select a particular pattern if necessary. The style is set with the command:

vsf_interior[vdi_handle, style];

where style is a value from zero through four. Zero selects a hollow style: the fill is performed in color zero, which is usually white. Style one selects a solid fill with the current fill color. A style of two is called "pattern" and a three is called "hatch", which are te somewhat suggestive of the options which can then be selected using the interior index. Style four selects the user defined pattern, which is described below.

The interior index is only significant for styles two and three. To set it, use:

vsf_style(vdi_handle, index);

(Be careful here: it is very easy to confuse this call with the one above due to the unfortunate choice of name.) The index selects the actual drawing pattern. The GEM VDI manual shows fill patterns corresponding to index values from 1 to 24 under style 2, and from 1 to 12 under style 3. However, some of these are implemented differently on the ST. Rather than try to describe them all here, I would suggest that you experiment. You can do so easily in ST Logo by opening the Graphics Settings dialog and playing with the style and index values there.

The user defined style gives you some interesting options for multi-color fills. It is set with:

vsf_udpat(vdi_handle, pattern, planes);

Planes determines the number of color planes in the pattern which you supply. It is set to one if you are setting a monochrome pattern. (Remember, monochrome is not necessarily black). It may be set to higher values on color systems: two for ST medium-res mode, or four for low-res mode. If you use a number lower than four under low-res, the other planes are zero filled.

The pattern parameter is an array of words which is a multiple of 16 words long. The pattern determined is 16 by 16 pixels, with each word forming one row of the pattern. The rows are arranged top to bottom, with the most significant bit to the left. If you have selected a multi-plane pattern, the entire first plane is stored, then the second, and so on.

Note that to use a multi-plane pattern, you set the writing mode to replace using vswr_mode(). Since the each plane can be different, you can produce multi-colored patterns. If you use a writing color other than black, some of the planes may "disappear".

Most of the solids Output Functions have analogous line drawing commands. The polyline command corresponds to the filled area primitive. The filled area routine is:

v_fillarea(vdi_handle, count, pxy);

Count and pxy are just the same as for v_pline(). If the polygon defined by pxy is not closed, then the VDI will force closure with a straight line from the last to the first point. The polygon may be concave or self-intersecting. If perimeter show is on, the area will be outlined.

One note of caution is necessary for both v_fillarea() and v_pline(). There is a limit on the number of points which may be stored in pxy[]. This limit occurs because the contents of pxy[] are copied to the intin[] binding array before the VDI is called. You can determine the maximum number of vertices by checking intout[14] after using the extended inquire function vq_extnd().

For reasons unknown to this writer, there are TWO different filled rectangle commands in the VDI. The first is

vr_recfl(vdi_handle, pxy);

Pxy is a four word array defining two opposite corners of the rectangle, just as in vs_clip(). Vr_recfl() uses the fill attribute settings, except that it NEVER draws a perimeter.

The other rectangle routine is v_bar(), with exactly the same arguments as vr_recfl(). The only difference is that the perimeter setting IS respected. These two routines are the fastest way to produce a solid rectangle using the VDI. They may be used in XOR mode with a BLACK fill color to quickly invert an area of the screen. You can improve the speed even further by turning off the clip (if possible), and byte aligning the left and right edges of the rectangle.

Separate commands are provided for solid circle and ellipse. The circle call is:

v_circle(vdi_handle, x, y, radius);

and the ellipse command is:

v_ellipse(vdi_handle, x, y, xrad, yrad);

All of the parameters are identical to those given above for v_arc() and v_ellarc(). The solid analogue of an arc is a "pie slice". The VDI pie commands are:

v_pieslice(vdi_handle, x, y, radius, begin, end);

for a slice from a circular pie, and

v_ellpie(vdi_handle, x, y, xrad, yrad, begin, end);

for a slice from a "squashed" pie. Again, the parameters are identical to those in v_arc() and v_ellarc(). The units and drawing order of angles are also the same. The final solids Output Function is:

v_rfbox(vdi_handle, pxy);

which draws a filled rounded rectangle. The pxy array defines two two opposite corners of the bounding box, as shown for vs_clip().

The issues involved in correctly setting the VDI attributes for a fill operation are identical to those in drawing lines. For those who want to employ the "always set" method, I have again included two skeleton routines in the download, which can be modified as desired.

TO BE CONTINUED

This concludes the first part of our expedition through basic VDI operations. The next issue will tackle the problems of drawing bit mapped text at a reasonable speed. This first pass will not attempt to tackle alternate or proportional fonts, or alternate font sizes. Instead, I will concentrate on techniques for squeezing greater performance out of the standard monospaced system fonts.

```
)))))))))))))))) Routines to set clip to a GRECT ((((((((((((((((((

        VOID
grect_to_array(area, array)     /* convert x,y,w,h to upr lt x,y and    */
        GRECT   *area;          /*                  lwr rt x,y          */
        WORD    *array;
        {
        *array++ = area->g_x;
        *array++ = area->g_y;
        *array++ = area->g_x + area->g_w - 1;
        *array = area->g_y + area->g_h - 1;
        }

        VOID
set_clip(clip_flag, s_area)     /* set clip to specified area           */
        WORD    clip_flag;
        GRECT   *s_area;
        {
        WORD    pxy[4];

        grect_to_array(s_area, pxy);
        vs_clip(vdi_handle, clip_flag, pxy);
        }

)))))))))) Routines to set attributes before output ((((((((((((

        VOID
rr_peria(mode, color, type, width, pxy)     /* Draw a rounded    */
        WORD    mode, color, width, *pxy;   /* rectangle outline */
        {
        vswr_mode(vdi_handle, mode);
        vsl_color(vdi_handle, color);
        vsl_type(vdi_handle, type);
        vsl_width(vdi_handle, width);
        v_rbox(vdi_handle, pxy);
        vswr_mode(vdi_handle, MD_REPLACE);
        }

        VOID
pl_peria(mode, type, color, width, npts, pxy)     /* Draw a polygonal */
                                                  /* figure           */
        WORD    mode, type, color, width, npts, *pxy;
        {
        vswr_mode(vdi_handle, mode);
        vsl_type(vdi_handle, type);
        vsl_color(vdi_handle, color);
        vsl_width(vdi_handle, width);
        v_pline(vdi_handle, npts, pxy);
        }

        VOID                    /* Draw a filled polygonal area */
pl_fill(mode, peria, color, interior, style, npts, pxy)
        WORD    mode, peria, color, interior, style, npts, *pxy;
        {
        vswr_mode(vdi_handle, mode);
        vsf_color(vdi_handle, color);
        vsf_style(vdi_handle, style);
        vsf_interior(vdi_handle, interior);
        vsf_perimeter(vdi_handle, peria);
        v_fillarea(vdi_handle, npts, pxy);
        }

        VOID                    /* Draw a filled rectangle    */
rect_fill(mode, peria, color, interior, style, pxy)
        WORD    mode, peria, color, style, interior, *pxy;
        {
        vswr_mode(vdi_handle, mode);
        vsf_color(vdi_handle, color);
        vsf_style(vdi_handle, style);
        vsf_interior(vdi_handle, interior);
        vsf_perimeter(vdi_handle, peria);
        vr_recfl(vdi_handle, pxy);
        }
```

I been waiting for YOU to write an Article for the MILATARI Newsletter

MILATARI BBS 414-781-5710 24 hours 300 Baud

From Page 2

MAKE LOWER CASE FILENAMES

Yes, fans, you CAN use lowercase letters for your disk filenames. You see that in MYDOS, right? Here's how you can do it with ANY DOS! If you look at the ATASCII codes, you'll see that the value of "0" (zero) is 48, and the value of lowercase "z" is 122. Do this:

1.) Boot in DOS with BASIC in.
2.) In the immediate mode type POKE 3818,48:POKE 3822,123 [RETURN].
3.) Type DOS and press [RETURN].
4.) Re-write DOS to your disk with the write DOS and DUP menu option.

These pokes set in the parameters to accept ATASCII values in the filenames starting from 48 and ending with 122 (but the 123 you see in the POKE above is correct). It's risky to go higher than 123, so just stick to what I told ya here. IT REALLY WORKS!!!

OPEN UP TO 7 FILES IN DOS

Have you ever noticed that you can have no more than three files open at one time in DOS? But ATARI allows you to have seven files, so why not also be allowed to have all seven of them to be opened at once? This is a free country, isn't it? Well, it ain't so tough to do. If you want to be able to open more than three files at once, just do this:

1.) Boot in DOS with BASIC in.
2.) In the immediate mode, type POKE 1801,7 [RETURN].
3.) Type DOS and press [RETURN].
4.) Re-write DOS and DUP to your disk with the DOS function menu.

A word of caution, though. Each file that you allow to be open uses a 128 byte data buffer. So you shouldn't open more files (or reserve more space) than you actually need. If, for example, you'll only need to open five files at once, then POKE 1801,5. See how it works?

(Another word of caution from the Editor. Be careful and try not to mix up the disks with the modified DOS with your normal versions. Both you and your friends may be confused and perplexed in the futur if you inadvertently spread these modified versions o. DOS around.)

MILWAUKEE AREA ATARI USER'S GROUP AND NEWSLETTER INFORMATION

MILATARI OFFICERS

President	Ron Friedel	354-1717
Vice President	Steve Tupper	462-8178
Secretary	Steve Armstrong	543-9039
Treasurer	Carl Mielcarek	355-3539

MILATARI VOLUNTEERS

MILATARI West	Dennis Bogie	968-2361
Education SIG	Joe Sanders	447-1660
ATR8000 & CP/M	Joe Griesemer	673-6663
BBS Sysop	Richard Dankert	781-2338
Membership	Dave Bogie	968-2361
Newsletter	Roy Duvall	363-8231
	Libraries	
Cassette &		
Copyright	Lee Musial	466-9160
Disk	Dennis Wilson	476-7767
	Bill Lawrence	968-3082
Publications	Bill Feest	321-4314

Milatari BBS

300 Baud 24hrs. 414-781-5710

NEWSLETTER INFORMATION

This newsletter is written and printed by members of the Milwaukee Area ATARI User's Group (MILATARI). Opinions expressed in this publication are those of the individual authors and do not necessarily represent or reflect the opinions of the Milwaukee Area Atari User's Group, its officers, its members or its advertisers except where noted. Articles reprinted from bulletin boards and other newsletters are presented as an information exchange, no warranty is made to their accuracy or factuality.

Your contributions of articles are always welcome. You may submit your article on ATARI compatible cassette or diskette, on typewritten form or you can arrange with the editor to upload your file via modem. You can send Graphics eight or seven plus screens stored on disk in Micropainter or Micro Illustrator formats.

Bored with the Newsletter? Write a review, an article.

Milwaukee Area Atari User's Group

MILATARI is an independent, user education group which is not affiliated with ATARI INC. The newsletter is the official publication of MILATARI and is intended for the education of its members as well as for the dissemination of information concerning ATARI computer products.

MILATARI membership is open to individuals and families who are interested in using and programming ATARI computers. The membership includes a subscription to this newsletter and access to the club libraries. The annual membership fee is $15 for individuals or $20 for a family.

Vendors wishing to display and/or sell items at MILATARI meetings must make prior arrangements with the club vice president. Rates are $10 per meeting or $90 per year payable in advance.

MILATARI ADVERTISING RATES

This newsletter will accept camera ready advertising copy from anyone supplying goods and services of interest to our membership.

Current paid members of MILATARI may place classified ads in the newsletter at no charge.

Advertising Rates

Full page	$37.50
Half page	$20.00
Quarter page	$12.50
Business card	$2.00

MILWAUKEE AREA ATARI USERS CLUB

U.S. POSTAGE PAID
BULK RATE
PERMIT NO. 201
WAUKESHA, WI 53187

MILWAUKEE AREA ATARI USERS GROUP
Post Office Box 19858
West Allis, Wisconsin 53219-0858

Don't be Piggish
with your knowledge
Share it in an article
in the MILATARI Newsletter